Curiel

Amaury

DL IM
GR8

RAPPORT DE PROJET :

COLT EXPRESS

I) Sources

Dans ce projet, je me suis aidé de plusieurs sources internet en particulier pour la construction de l'interface graphique.
Je me suis aidé de :
La chaine YouTube de Dominique Liard :
https://www.youtube.com/watch?v=SQae_yJBK1c&t=2s

Du site internet :
https://www.jmdoudoux.fr/java/dej/chap-swing.htm
Du site internet:
https://baptiste-icht.developpez.com/tutoriels/java/swing/debutant/

Enfin, pour gérer l'affichage des wagons du projet, j'ai utiliser la classe Panel Image :

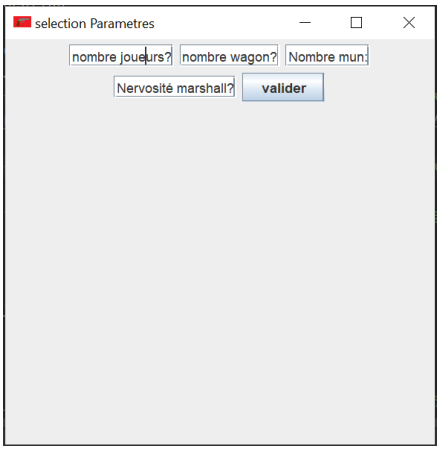
https://codes-sources.commentcamarche.net/source/54144-afficher-une-image-en-arriere-plan-dans-un-jpanel

codée par « cs_Julien39 ».

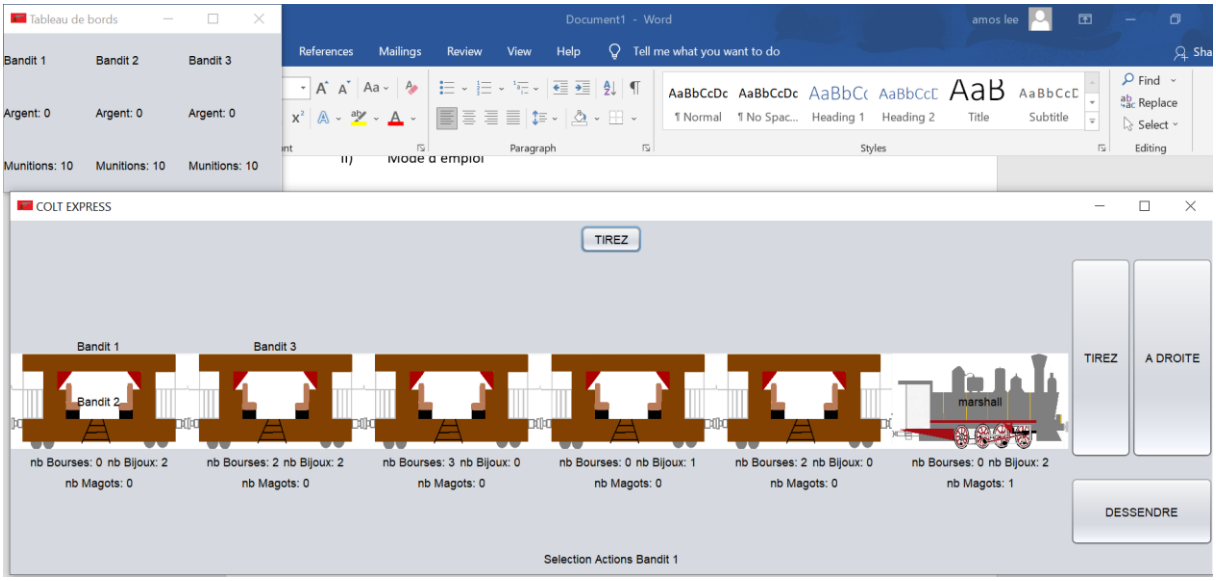
En plus de ces sources, je me suis aussi aidé de divers forums pour comprendre les mécanismes tout en mettant en œuvre moi-même les idées.

II) Mode d'emploi

Lors du lancement de l'application, une boite de dialogue s'ouvre et demande les valeurs souhaitées pour les différents paramètres.



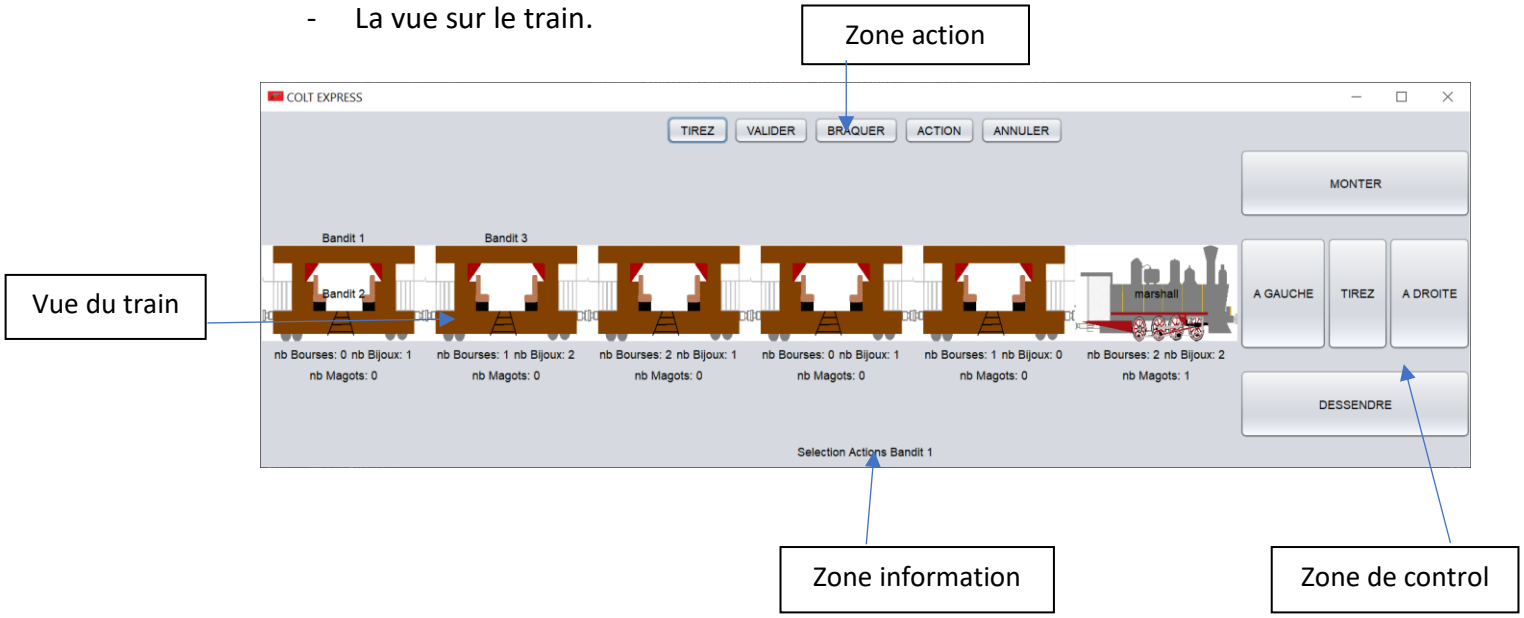
Bien que cela fonctionne pour toutes les valeurs entières, il est conseillé de mettre au moins 3 wagons affin que l'image ne sois pas trop déformée. Il est aussi conseiller de mettre au maximum 10 wagons si vous avez un écran de taille standard.
Une fois que vous avez cliquer sur le bouton valider, l'application s'ouvre enfin avec une fenêtre « Tableau de bord » qui contient les informations sur les bandits.



Les bandits se répartissent en occupant le toit des wagons pour les bandits impairs et l'intérieur pour les bandits pairs. Ainsi, le Bandit 1 occupera le toit du wagon numéro 1, le Bandit 3 le toit du wagon numéro 2 et les bandits 2 et 4 occuperont l'intérieur des wagons 1 et 2.
L'application comprends plusieurs zones :

- Une zone d'action contenant les boutons « Braquage », « Action », « Valider » …
- Une zone de control contenant les boutons servant à diriger le bandit

- Une zone d'information chargée d'informer les joueurs sur l'état de leur bandit et de les informer sur le déroulement de la partie (Compte rendu)
- La vue sur le train.



(Cette image n'est pas obtenable en lançant l'application telle qu'elle et est là qu'à titre d'exemple)

Le but du jeu est de braquer touts les wagons du train. Lorsque cet évènement est atteint, le jeu s'arrête. Et le nom du vainqueur est affiché dans la zone d'information.

III) Architecture de l'application

Pour fonctionner, l'application utilise 15 classes. Je vais spécifier ce que fais chacune d'entre elle. Vous trouverez en annexe un diagramme de classe.

1) La classe Colt Express
   Cette classe est la classe principale de l'application, elle contient seulement la méthode main qui se contente d'afficher la fenêtre de dialogue et d'afficher la fenêtre principale de l'application.
2) La classe InterfaceGraphique
   Cette classe est la classe qui gère la fenêtre principale de l'application. Elle possède 2 attributs de type int chargés de compter le nombre de tour et de savoir quel joueur joue.
   Elle possède également 3 conteneurs de type JPanel contenant respectivement le contrôleur, la barre d'action ainsi que la barre d'information.
   Elle possède enfin un attribut de type Train chargé de l'affichage du train ainsi qu'un objet de type Constantes chargé de mémoriser touts les paramètres de l'application.

Le constructeur de la classe est chargé d'initialiser les variables entières ainsi que la constante. Elle construit également le contrôleur la barre d'info et la barre d'action.
La classe possède également une méthode « calculBouton » charger d'afficher les boutons possibles pour le bandit en cours.
Enfin, la classe implémente la méthode actionPerformed chargée de répondre de manière adéquate au clic de l'utilisateur.

3) La classe Train

La classe Train est chargée de l'affichage du train.
Plusieurs attributs de type JPanel chargé d'accueillir les sprites des Wagons ainsi que les différents JLabels nécessaires.
Le constructeur de la classe construit un train sous la forme d'un GridLayout composé de 3 lignes et d'autant de colonnes que de Wagons contenant chacun un layout de type BorderLayout.
La classe possède également une méthode createTresature() chargée de disposer les trésors dans les wagons, une méthode createLabel chargée de créer les JLabels et les mettre dans un tableau, une méthode updateLabel() chargée de mettre a jour les labels et une methode getNBTresor() utiliser pour determiner la fin du jeu.

4) La classe Wagon

La classe Wagons est juste là pour savoir combien de bourses, bijoux et magot est présent dans le wagon. Le constructeur est chargé de disposer les bijoux, magots dans le wagon. Elle possède également des getters ainsi que des setters.

5) La classe Constantes

Comme mentionné précédemment, la classe Constantes contient tous les paramètres utiles à l'application. Elle possède que des attributs static ce qui permets de pouvoir y accéder depuis n'importe quelle classe.

6) Le type énumérer Direction

Il s'agit d'un type énumérer listant toutes les possibilités pour les actions des bandits.

7) Le type Bourse

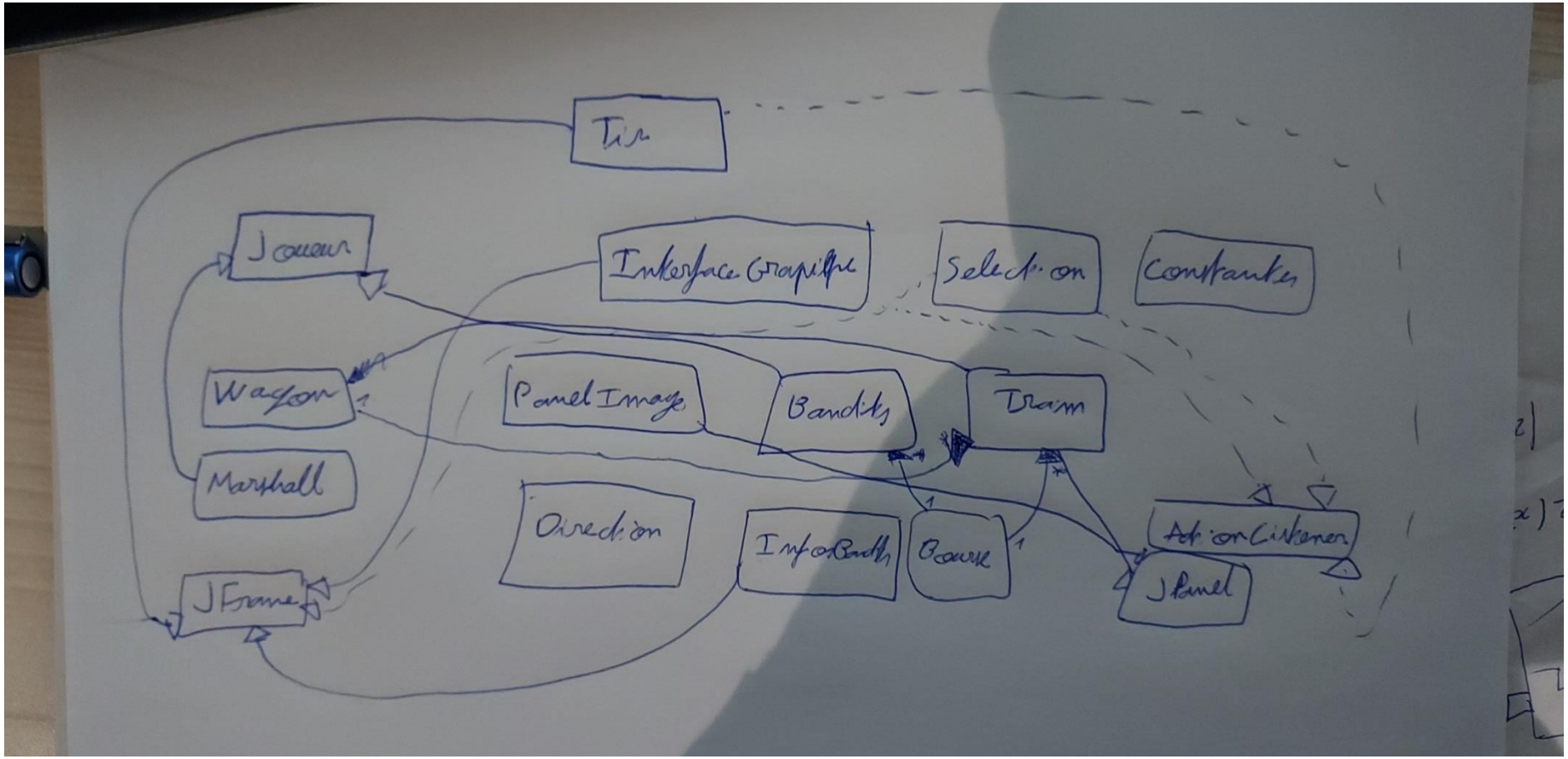
Le type bourse est un type qui sers seulement a stocker la valeur de la bourse.

8) Les fenêtres Tir, Selection et InfosBandits
   Il s'agit de fenêtres s'ouvrants quand on effectue certaines actions. Toutes contiennent des JFrames.

9) La classe abstraite Joueur
   Cette classe modélise le joueur (Bandits ou Marshall). Elle contient son nom, son indice, son wagon ainsi que son étage. Son constructeur initialise le joueur en faisant en sorte que sa position initiale soit telle que décrite dans le II).
   Elle possède une fonction qui permets de déplacer le joueur, une methode setAction qui permets de remplir le tableau des actions du joueur et une fonction tirer.
   Elle possède également les fonctions getCurrantW/E charger de calculer le wagon/etage du joueur suite aux actions qu'il vient d'effectuer.
10) La classe Bandits
   Un bandit est un joueur. IL possède en plus un butin et un nombre de munition. Il possède en plus de cela une fonction braque chargée de braquer un wagon.
11) La classe Marshall
   Un Marshall est un joueur doté d'une fonction deplaceM qui lui permets de préremplir ces déplacements ainsi que d'une fonction stringAction qui lui permets de convertir ces actions en phrases.

12) La classe Panel Image
   / ! \ La classe panel image n'est pas implémentée par moi-même comme mentionné dans I)
   La classe Panel Image est chargée de pouvoir afficher une image en arrière-plan dans un JPanel.

IV) Diagrame de classe


Tir
Joueur
Interface Graphique
Selection
Contrainte
Wagon
Panel Image
Bandits
Train
Marshall
Direction
InfoBandit
Course
ActionListener
JFrame
JPanel